REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	12 Jan 2006	Final Report 15 Oct 2004-14 June 2005

4. TITLE AND SUBTITLE
Autonomous Formation Flight for Autonomous and Semi-Autonomous Rotorcraft Using MPC

5. FUNDING NUMBERS
W911NF-04-1-0448

6. AUTHOR(S)
Hoam Chung and S. Shankar Sastry

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of California
336 Sproul Hall
Berkeley, CA 94720

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U. S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
47547.1-CI-II

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12 a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12 b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Formation flight is the primary movement technique for teams of helicopters. However, the potential for accidents is greatly increased when helicopter teams are required to fly in tight formations and under harsh conditions. This project proposed that automation of helicopter formations is a realistic solution capable of alleviating risks. Our ultimate research objective is to develop technologies that work for both manned and unmanned rotorcraft, since it is our sense that they will be working together in the future. We also deeply believe that it is important to develop tools for human-centered automation as a via-point to complete automation starting point is to design a distributed ontrol law attenuating external distrubances coming into a formation, so that each vehicle can safely maintain sufficient space between it and all other vehicles. While conventional methods are limited to homogeneous formations, our decentralized MPC approach allows for heterogeneity in formation.

14. SUBJECT TERMS	15. NUMBER OF PAGES
helicopter formations, autonomous control, formation manager, model-predictive control, heterogeneous formations, human-centered automation	44
	16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT	18. SECURITY CLASSIFICATION ON THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UU

ARO STIR W911NF-04-1-0448 Final Report

Autonomous Formation Flight for Autonomous and Semi-Autonomous Rotorcraft Using MPC

Berkeley Aerobot (BEAR) Program

January 2006

Shankar Sastry

University of California, Berkeley

Contents

1 Statement of Problem: Automation of Helicopter Formations

Formation flight is the primary movement technique for teams of helicopters. However, the potential for accidents is greatly increased when helicopter teams are required to fly in tight formations and under harsh conditions. ARO STIR W911NF-04-1-0448 proposed that the automation of helicopter formations is a realistic solution capable of alleviating risks. Helicopter formation flight operations in battlefield situations are highly dynamic and dangerous, and, therefore, we maintain that both a high-level formation management system and a distributed coordinated control algorithm should be implemented to help ensure safe flight formations.

Our ultimate research objective is to develop technologies that work both manned and unmanned rotorcraft, since it is our sense that they will be working together in the future. We also deeply believe that it is important to develop tools for human centered automation as a via-point to complete automation. As a preliminary work toward autonomous (for unmanned rotorcraft) and semi-autonomous (for manned helicopters) formation flight control and management system, we designed decentralized nonlinear model predictive control (MPC) law for coupled systems. Also, a hardware-in-the-loop simulation (HILS) system was implemented as the future development tool.

The starting point for ARO STIR W911NF-04-1-0448 was to design a distributed control law attenuating external disturbances coming into a formation, so that each vehicle can safely maintain sufficient space between it and all other vehicles. While conventional methods are limited to homogeneous formations, our decentralized MPC approach allows for heterogeneity in a formation. In order to avoid the conservative nature inherent in distributed MPC algorithms, we begin by designing a stable MPC for individual vehicles, and then introducing carefully designed inter-agent coupling terms in a performance index. Thus the proposed algorithm works in a decentralized manner, and can be applied to the problem of safe helicopter formations comprised of heterogenous vehicles.

In order to perform formation flight experiments based on the proposed algorithm while minimizing the possibility of software failure, we developed a hardware-in-the-loop simulation (HILS) system which is compatible with our rotorcraft-based unmanned aerial vehicle (RUAV) systems. Using multiple HILS system enables us to verify newly developed code in realistic circumstances. Also, the developed HILS system will be used as a virtual

helicopter in the future experiments.

Section 2 describes linear helicopter dynamics and nonlinear kinematics, and a cruise model of the Yamaha R-50 industrial unmanned helicopter [23], which is used throughout this report. We introduce a scaled-up model by taking advantage of dimensional analysis for the heterogeneous formation used in Section 3. In Section 3, the MPC algorithm for autonomous helicopter formations is formulated. Decoupled MPC algorithms for individual vehicles are designed based on the Control Lyapunov Function (CLF) approach [17]. The formation topology is defined, and the varying gap strategy and the constant gap strategy are introduced. Through computer simulations, we show that the proposed MPC scheme successfully attenuates external disturbance propagation even in a heterogeneous formation. Section 4 describes the hardware and software structure of the HILS system for Berkeley Aerobot (BEAR) fleet. A summation of work completed on ARO STIR W911NF-04-1-0448 is given in Section 5 followed by conclusions and suggestions for future work.

2 Helicopter Dynamics

Helicopter dynamics with a nonlinear kinematics model in a general form is presented. Based on this, the combined dynamics and kinematics model for the forward cruise flight mode is derived in the section 2.2.

We need a set of heterogeneous helicopter cruise models for simulations, since our algorithm proposed in the next section is focused on the disturbance attenuation property in a heterogeneous formation. Because a cruise model of a helicopter with proper size is extremely rare in the literature, we simply use a dimensional analysis technique presented in [23] to generate a scaled-up virtual model. Details of how the model is scaled up are discussed in Section 2.3.

2.1 Helicopter Dynamics with Nonlinear Kinematics

Since the helicopter dynamics, which can be derived using Newton's law, are represented in the body coordinates system fixed to the center of the mass of a helicopter [32], the kinematic equations between the body coordinates and the spatial coordinates[1] are required. The kinematics are further divided into two parts: (1) the position describing translational motion in the spatial

[1]Throughout this report, the spatial coordinates mean the tangent-plane coordinate system, whose origin is located at a certain point of the earth's surface.

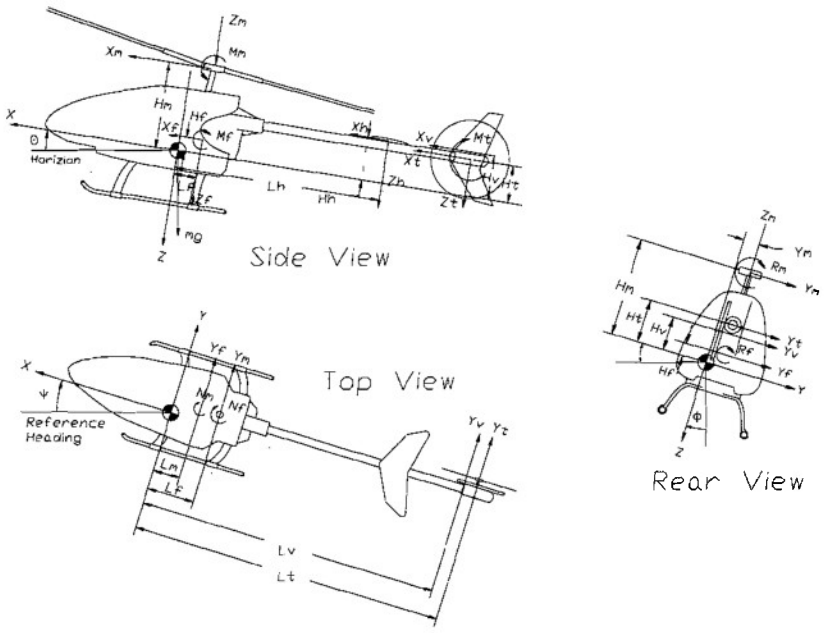


Figure 1: Free body diagram of a helicopter [32]

coordinates, and (2) the Euler angles describing the vehicle's attitude and heading in the spatial coordinates.

The following definitions of helicopter dynamics and kinematics are based on [32, 34] with slight modifications. Figure 1 shows the body-fixed coordinates, forces and moments exerted on the helicopter. The positive directions of x, y, and z axes of the spatial coordinates align, respectively, to the north, east and downward directions. For detailed derivations of the helicopter dynamics, see [32] and [23].

As mentioned earlier, the overall system dynamics are divided into the kinematics and the system-specific dynamics denoted by superscripts K and D. The state vectors and the control input vector are defined as follows:

$$\mathbf{x}^D = \begin{bmatrix} u & v & p & q & a_{1s} & b_{1s} & w & r & r_{fb} & c & d \end{bmatrix}^T \tag{1}$$

$$\mathbf{x}^A = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}, \quad \mathbf{x}^K = \begin{bmatrix} \mathbf{x}^A \\ \mathbf{x}^S \end{bmatrix}, \tag{2}$$

$$\mathbf{u} = \begin{bmatrix} u_{a1s} & u_{a1s} & u_{\theta_M} & u_{ref} \end{bmatrix}^T, \tag{3}$$

where

u, v, w: trimmed translational velocities in body coordinates

p, q, r: roll, pitch and yaw rates in the body coordinates

ϕ, θ, ψ: denote roll, pitch, and yaw in ZYX Euler angle notation[2] in the spatial coordinates

a_{1s}, b_{1s}: longitudinal and lateral flapping angles of the main rotor

c, d: longitudinal and lateral flapping angles of the Bell-Hiller stabilizer bar

r_{fb}: internal state of feedback gyro

u_{a1s}, u_{a1s}: inputs to the lateral and longitudinal cyclic pitch

u_{θ_M}: input to the main rotor collective pitch

u_{ref}: reference yaw rate input to the gyro

Let the superscripts S and B denote spatial and body coordinates. $\mathbf{x}^S$ and $\mathbf{x}^B$ denote the position in the spatial coordinates and in the body coordinates, respectively[3].

The kinematics part can be defined as follows:

$$\dot{\mathbf{x}}^S = R^{B\to S}\dot{\mathbf{x}}^B, \quad \dot{\mathbf{x}}^A = R_\omega^{B\to S}\omega, \tag{4}$$

where $\mathbf{x}^S = [x^S\ y^S\ z^S]^T$, and $\omega = [p\ q\ r]^T$. $R^{B\to S}(\mathbf{x}^A)$, the rotation matrix from the body to the spatial coordinates, and $R_\omega^{B\to S}(\mathbf{x}^A)$ is the relationship matrix between angular rates in the body and in the spatial coordinates. They are defined by [6, 20]

$$R^{B\to S} = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{5}$$

$$R_\omega^{B\to S} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix}. \tag{6}$$

[2]Rotate ψ in Z, θ in Y' and ϕ in X''. Note that this transformation results in same transformation with XYZ fixed angles [6]

[3]subscript vehicle indexes are suppressed until the next section for simplicity

The dynamics part can be written as

$$\dot{\mathbf{x}}^D = f^D(\mathbf{x}^D(t), \mathbf{x}^A(t), \mathbf{u}(t)). \tag{7}$$

Finally, the entire system equation becomes

$$\frac{d}{dt}\begin{bmatrix}\mathbf{x}^D \\ \Theta \\ \mathbf{x}^S\end{bmatrix} = \begin{bmatrix}f^D(\mathbf{x}^D(t), \mathbf{x}^A(t), \mathbf{u}(t)) \\ R_\omega^{B\to S}\omega \\ R^{B\to S}\dot{\mathbf{x}}^B\end{bmatrix} = f(\mathbf{x}^S(t), \mathbf{x}^D(t), \mathbf{x}^A(t), \mathbf{u}(t)), \tag{8}$$

or simply,

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \quad \text{with} \quad \mathbf{x} \triangleq \begin{bmatrix}\mathbf{x}^D \\ \mathbf{x}^A \\ \mathbf{x}^S\end{bmatrix}. \tag{9}$$

2.2 Helicopter Cruise Model

In [23], a linear cruise flight model of the Yamaha R-50 industrial helicopter trimmed at $u_0 = 49$ ft/sec, $w_0 = 11.2$ ft/sec, and $v_0 = 0$ is introduced and all the coefficients in the dynamics equation are identified through test flights and system identification techniques. If we convert this trim condition into spatial coordinates, then it becomes 30 mi/h forward cruise speed with the pitch trim $\theta_0 = -0.22$(rad).

In order to use the linear cruise model, we need to define several relationships between spatial variables and variables in the linear dynamics. First, the velocities in the body-fixed frame can be represented by

$$\dot{x}^B = u_0 + u, \ \dot{y}^B = v_0 + v, \text{ and } \dot{z}^B = w_0 + w. \tag{10}$$

Next, the trimmed pitch angle $\bar{\theta}$ can be defined as

$$\bar{\theta} = \theta - \theta_0. \tag{11}$$

From these relationships, the kinematics equations (4) are now well defined. The dynamics can be represented by

$$\dot{\mathbf{x}}^D(t) = f^D(\mathbf{x}^D(t), \mathbf{x}^A(t), \mathbf{u}(t)) = A\mathbf{x}^l(t) + B\mathbf{u}(t), \tag{12}$$

where $A \in \mathbb{R}^{11\times 13}$, $B \in \mathbb{R}^{11\times 4}$, and

$$\mathbf{x}^l = \begin{bmatrix} u & v & p & q & a_{1s} & b_{1s} & w & r & r_{fb} & c & d & \phi & \bar{\theta}\end{bmatrix}^T. \tag{13}$$

The helicopter hovering mode is known to be an unstable equilibrium, which means that any small perturbation will cause the vehicle to diverge from the hovering condition. On the contrary, the helicopter dynamics in the forward flight condition are stable with respect to a given pitch angle, and converge to a fixed point in the state space.

Table 1: Comparison of Froude numbers

	Rotor Radius (ft)	Rotor Speed (rad/s)	N	Froude Number
R22	13	53	0.38	1134
Virtual Model	10	62	0.5	1194
R-50	5	89	1	1230

2.3 Scaling Based on Froude Number

In order to test the algorithm we propose in the next section with a heterogeneous helicopter team, we needed to generate a model that has different dynamics characteristics from our existing Yamaha R-50 model. Since it is extremely difficult to perform an identification flight in a cruise condition without a wind tunnel facility, we used the scaling technique presented by Mettler [23].

The Froude number is the ratio of inertia to gravitational forces. If two different models have Froude numbers that are close each other, it means that two systems have similar dynamic properties. The number is defined by

$$F = \frac{V^2}{Lg}, \tag{14}$$

where V is the characteristic velocity, L is the characteristic length, and g is gravitational acceleration. In helicopter dynamics, the rotor tip velocity and the rotor radius are used as V and L respectively. The Table 1 shows the comparison of the Froude numbers of Yamaha R-50 and Robinson R22, and they are very close. We have created a virtual model in the region between the Robinson R22 and Yamaha R-50, which has two twice the rotor diameter of the R-50 and has a similar Froude number. The scale N refers to a model helicopter with $1/N$ the rotor diameter of the Yamaha R-50. It should be noted that the relationship of the Froude number imposes a relation between time scales [23],

$$\sec \approx \frac{1}{\sqrt{N}}. \tag{15}$$

Based on these comparisons, the scaling of coefficients in A and B (12) can be done using dimensional analysis.

Figure 2 and 3 shows frequency responses of the R-50 cruise model and the scaled-up virtual model. As expected, the attitude dynamics of the virtual model are much slower than those of the R50. This fact can be

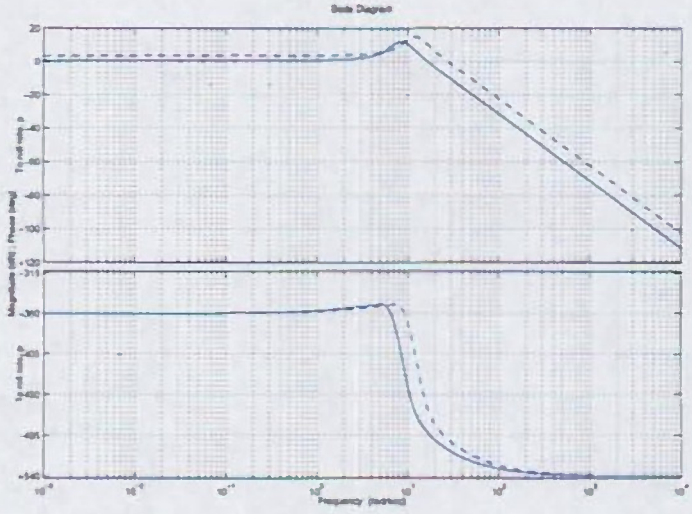


Figure 2: Frequency response from lateral cyclic pitch u_1 to body roll rate p (solid: virtual model, dashed: R-50 model)

reconfirmed by step responses of attitude and body-velocity dynamics (Figure 4).

3 Summary of Results: MPC-Based Helicopter Formation Flight

Model predictive control (MPC), also known as moving horizon or receding horizon control (RHC), has been a useful technique for the control of slow dynamical systems such as chemical processes because such a scheme requires high computational speed of the control hardware due to its online nature. With the rapid development of digital processors, powerful and inexpensive controllers make it possible to adopt MPC in hard real-time applications [31].

MPC can provide a better performance in controlling uncertain plants since it can update the gain of the controller based on the current states, while fixed-gain control algorithms cannot [21]. The capability to manipulate the state-dependency of the control weighting matrices and constraints in real-time is the key feature of a model predictive control algorithm. There are excellent survey papers describing the development of MPC theories. See

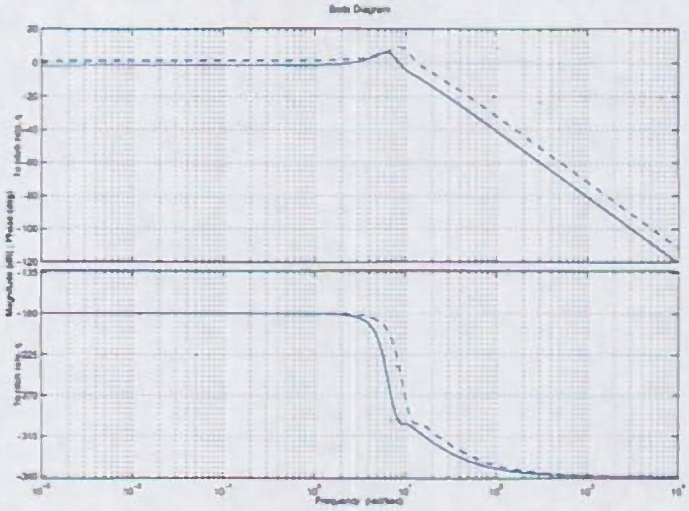


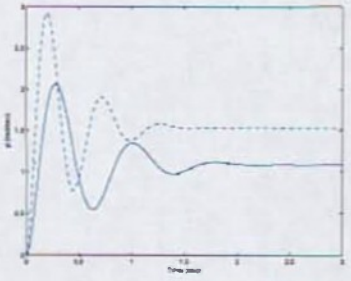
Figure 3: Frequency response from longitudinal cyclic pitch u_2 to body pitch rate q (solid: virtual model, dashed: R-50 model)

[22] and [3] for example.

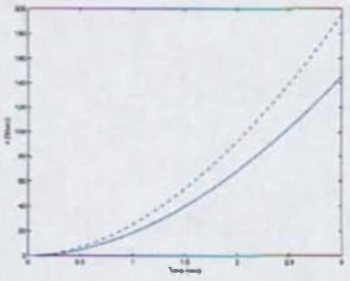
As long as an MPC algorithm is applied to a formation flight problem, a centralized approach is not a feasible choice at all, since it is not scalable from the viewpoint of computation and communication [7].[4] Thus it is natural to consider a decentralized approach to the formation flight problem. However, with a decentralized MPC scheme, it is recognized that the stability proof becomes very difficult.

Under the assumption that the dynamics of each vehicle are decoupled, a major obstacle in proving the stability of a decentralized MPC scheme arises in predicting neighbors' behaviors over the future horizon. Without considering inter-vehicle constraints, the coupling between agents appears in the performance index as a penalty on relative gap errors. If there is no appropriate predictions of the behaviors of neighbors, it is difficult to set bounds on them. In an attempt to resolve this difficulty, authors of [7] introduced so called 'compatibility constraints', which restrict the future variations of neighbors' optimal inputs from the previous optimal ones. Using this, it can

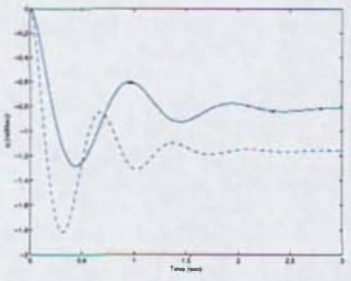
[4] In terms of communication, the amount of communication in a decentralized setup can be more than that of a centralized version depending on the type of numerical procedures for achieving an optimal solution. This subject is discussed in Section 3.4 more detail.



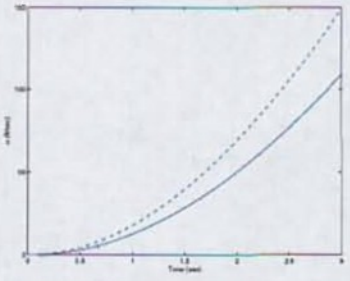
(a) roll rates in body coordinates, lateral cyclic step input



(b) lateral velocities in body coordinates, lateral cyclic step input



(c) pitch rates in body coordinates, longitudinal cyclic step input



(d) longitudinal velocities in body coordinates, longitudinal cyclic step input

Figure 4: Step response comparisons between the R-50 (dashed) and the virtual model (solid)

be proved that the closed-loop states converge to the neighborhood of origin. However, due to the nature of this constraint, once an open-loop control is computed and applied to the system on the current sampling time, the control at the next sampling time is constrained by the previous open-loop control. In nominal situations (no model errors without exogenous disturbances), this may not be a problem, since the open-loop control will predict the system behaviors exactly and the system may stay on the optimal trajectory. However, if the system trajectory starts to deviate from the initial optimal trajectory because of disturbances or model errors, this constraint would limit the effect of feedback, and the robust nature of the feedback system might be lost. Before the stability is affected by uncertainties, the algorithm may have trouble maintaining the feasibility of the optimization problem. Furthermore, since this algorithm is applicable only to (nonlinear) double integrator systems, it is impossible to use this algorithm for formation flight of helicopters, which have unmeasurable hidden state variables related to flapping and stabilizer bar dynamics.

In [19], researchers used the hierarchical decomposition method, which decomposes the original formation graph into overlapping subgraphs with different hierarchy levels. Under this decomposition, the algorithm allows a vehicle at a node with high priority to compute control laws for vehicles with lower priorities, and transmit them to vehicles with lower priorities, assuming a one time step communication delay. By doing this, since future behaviors of neighboring vehicles with lower priorities are completely known to vehicles with higher priorities, 'prediction' is no longer needed, and stability can be proved by standard Lyapunov arguments. In theory, this method provides a simple and clear way to prove the stability of a decentralized MPC scheme ,minimizes the conservatism and required communication bandwidth. However, in this case, the integrity of the system structure is totally dependent on the communication link, which can be deteriorated easily in battlefields.

Instead of sticking to proving stability of a decentralized MPC, our focus is on designing an MPC-based velocity tracking controller with penalties on relative gap errors, and study the propagation of external disturbances through the formation. In the following sections, we suggest two strategies of defining relative gap errors between neighboring vehicles, and compare their disturbance attenuation performance in three dimensional helicopter formation simulations.

3.1 Formation Topology and Definitions of Gap Errors

In the following discussion, we examine formations where each agent in the formation has connections that are less than or equal to two. Although cases where one or more agents in the formation has more than three connections can still be accounted for, these are considered special cases, and are not described here. In real-world operations, most helicopter formations fall into the category with a maximum two bidirectional connections on each agent [15]. Some publications [8, 19] on the vehicle formations using distributed MPC algorithms consider arbitrary formation shapes represented by connected graphs. However, an arbitrary formation shape is obviously not used in high-speed cruise formations, especially for manned helicopters. We believe that the research on arbitrarily coupled vehicle formation needs to be developed in the context of behaviors of a 'swarm' of unmanned vehicles [1].

Most vehicle formation algorithms [8, 19, 26, 29] use a so-called 'constant gap' strategy as described in Figure 5. $l^r_{i-1,i} \in \mathbb{R}^3$ denotes the constant relative gap vector from $i-1$-th vehicle to i-th vehicle in the reference coordinates, which is represented by $x^r - y^r$ (tangential-normal to the reference velocity V_r^S) in the figure. Note that we need the relationship

$$l^r_{i-1,i} = -l^r_{i,i-1} \tag{16}$$

for the unique definition of the formation shape. Also, we can obtain the $l^S_{i-1,i}$ using the rotation matrix $R^{r\to S}(\psi_r)$ such that

$$l^S_{i-1,i} = R^{r\to S}(\psi_r) l^r_{i-1,i}. \tag{17}$$

As shown in Figure 5, the gap error can be defined as

$$\begin{aligned} \mathbf{e}^S_{i,i-1} &= \mathbf{x}^S_{i-1} - \mathbf{x}^S_i + l^S_{i-1,i} \\ &= \mathbf{x}^S_{i-1} - \mathbf{x}^S_i - l^S_{i,i-1} \end{aligned} \tag{18}$$

The other type of the gap strategy is called a 'varying gap' strategy (Figure 6). In this strategy, the i-th vehicle considers the middle point in the line connecting $i-1$-th vehicle to $i+1$-th vehicle as the reference point. The error vector becomes

$$\mathbf{e}^S_i = \frac{\mathbf{x}^S_{i-1} + \mathbf{x}^S_{i+1}}{2} - \mathbf{x}^S_i. \tag{19}$$

For vehicles in edges, the constant gap strategy (18) is used, although other vehicles use the varying gap strategy. By using the constant strategy in edges, the sum of squares of gap errors is zero if and only if all the gap errors are zero, even if we use the varying gap strategy.

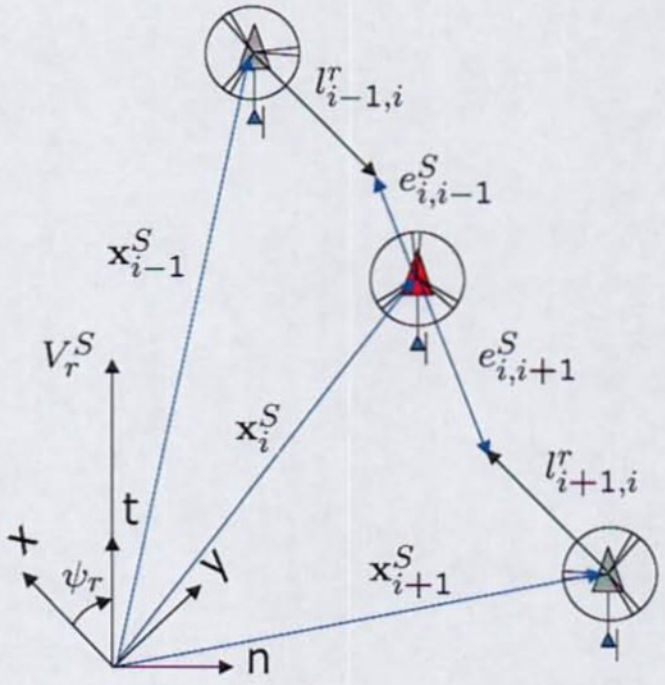


Figure 5: Error vector definition, constant reference gap

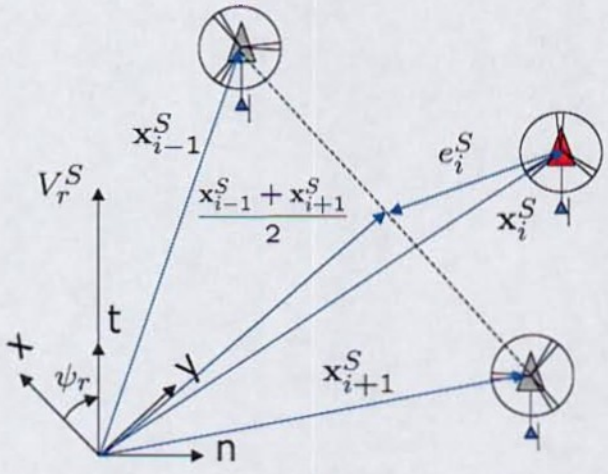


Figure 6: Error vector definition, varying reference gap

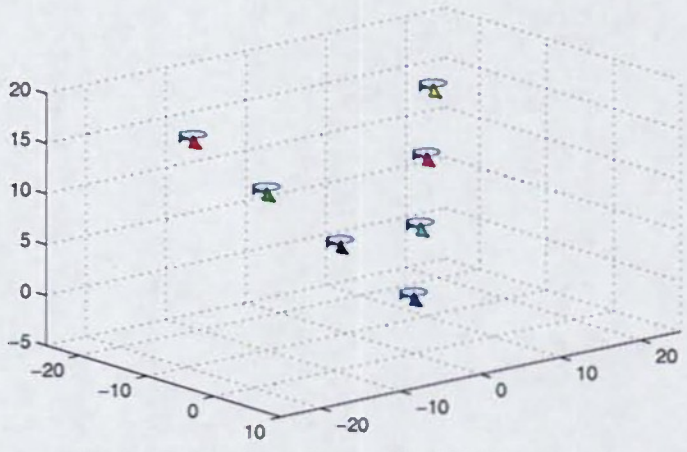


Figure 7: Configuration of the 3D Vee formation

The definition of the error vector also varies according to the role of a vehicle in the formation, e.g., leader vehicle. Suppose that a vehicle with index i is leading a three-dimensional Vee formation (Figure 7). Then, the objective of the leader is to maintain constant gaps in x^r and z^S direction from the nearest neighbors, while it uses the varying gap strategy in y^r direction. Let us define indices of the nearest neighbors α_i^x and α_i^z such that

$$\begin{aligned} \alpha_i^x &= \begin{cases} i-1 & \text{if } |x_i^r - x_{i-1}^r| \leq |x_i^r - x_{i+1}^r| \\ i+1 & \text{otherwise} \end{cases} \\ \alpha_i^z &= \begin{cases} i-1 & \text{if } |z_i^S - z_{i-1}^S| \leq |z_i^S - z_{i+1}^S| \\ i+1 & \text{otherwise} \end{cases} \end{aligned} \tag{20}$$

The gap error vector for the leader is finally defined as

$$\mathbf{e}_i^r = \begin{bmatrix} x_{\alpha_i^x}^r - x_i^r - l_{i,\alpha_i^x}^r|_x \\ \frac{1}{2}(y_{i-1}^r + y_{i+1}^r) \\ z_{\alpha_i^z}^r - z_i^r - l_{i,\alpha_i^z}^r|_z \end{bmatrix}, \tag{21}$$

where $l|_{\{x,y,z\}}$ denotes the corresponding element of a vector l. The gap error vector for the leader is illustrated in Figure 8. Note that, although the constant gap strategy is used in x^r and z^r directions, the overall definition of the gap error vector is time-varying, since α_x^i and α_z^i are not constant with respect to time.

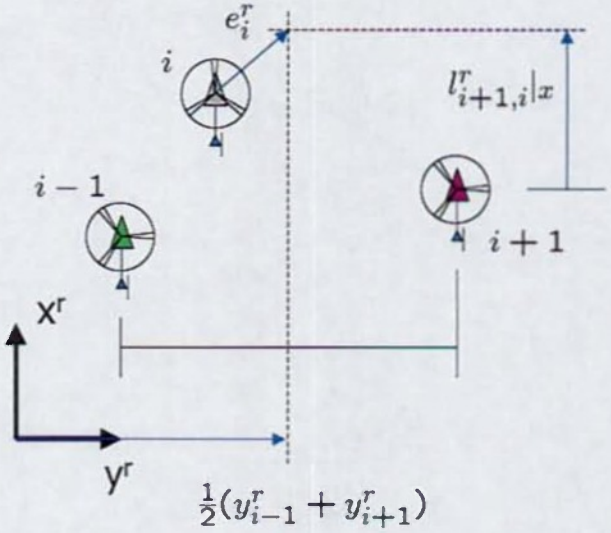


Figure 8: Illustration of error definition for a leader vehicle

By using gap error vector definitions represented above, we can implement most of real-world helicopter formations. For example, Vee, wedge, left and right echelon, and left and right staggered formations [15] can be realized.

3.2 Model Predictive Control Law for Helicopter Formations

Recall the system equation of the i-th vehicle

$$\dot{\mathbf{x}}_i(t) = f_i(\mathbf{x}_i(t), \mathbf{u}_i(t)), \tag{22}$$

where

$$\mathbf{x}_i \in \mathcal{X}_i \subset \mathbb{R}^{n_i^x}, \ \mathbf{u}_i \in \mathcal{U}_i \subset \mathbb{R}^{n_i^u}. \tag{23}$$

Here $\mathcal{X}_i$ and $\mathcal{U}_i$ are convex sets. We assume that $\mathbf{u}_i(t)$ is measurable, and $f_i : \mathcal{X}_i \times \mathcal{U}_i \to \mathbb{R}^{n_i^x}$ satisfies standard conditions for admitting a unique solution. Remember previously defined this nonlinear helicopter model as a combination of six degree-of-freedom nonlinear kinematics and linear forward cruise dynamics in Section 2.

The performance index $J_i(\cdot)$ has the form of

$$J_i(\bar{\mathbf{x}}_i, \kappa_i, t) = \int_{\tau}^{\tau+L} \mathcal{L}_i(\mathbf{x}_i(t), \mathbf{x}_{-i}^S(t), \mathbf{u}_i(t), \mathbf{y}_i^S(t))dt + V_i^f(\mathbf{x}_i(\tau+L), \mathbf{x}_{-i}^S(\tau+L), \mathbf{y}_i^S(\tau+L)), \tag{24}$$

where the terminal penalty function $V_i^f(\cdot) : \mathcal{X}_i \times \mathcal{X}_{-i} \times \mathbb{R}^{n_i^y} \to \mathbb{R}$ is positive definite. The subscript $-i$ denotes indices of neighbors of the i-th vehicle. $\mathbf{y}_i^S : \mathbb{R} \to \mathbb{R}^{n_i^y}$ is the reference vector in the spatial frame, which has (at least) the reference velocity vector $\dot{\mathbf{x}}_i^{S,r}(t)$ and the reference heading $\psi_i^r(t)$. The running cost $\mathcal{L}_i : \mathcal{X}_i \times \mathcal{X}_{-i} \times \mathcal{U}_i \times \mathbb{R} \to \mathbb{R}_+$ has the property that $\mathcal{L}_i(0,0,0,0) = 0$.

The finite horizon optimal control (FHOC) problem with initial condition $\bar{\mathbf{x}}_i = \mathbf{x}_i(\tau)$ and horizon length L is defined as

$$V_i(\bar{\mathbf{x}}_i, t) = \min_{\kappa} J_i(\bar{\mathbf{x}}_i, \kappa_i, t), \tag{25}$$

which is subject to (22), and (23).

κ_i is a piecewise continuous time-dependent function in open-loop strategy space such that

$$\kappa_i \in \mathcal{K}_i = \{\kappa : [0, L] \times \mathcal{X}_i \to \mathcal{U}_i\} \tag{26}$$

$$\mathbf{u}_i(t) = \kappa_i(t - \tau, \bar{\mathbf{x}}_i). \tag{27}$$

If an optimal solution of the FHOC problem exists, let $\kappa^*(t - \tau, \bar{\mathbf{x}}_i)$ denote the solution for $t \in [\tau, \tau + L]$. Note that $V_i(\bar{\mathbf{x}}_i, t) = J(\bar{\mathbf{x}}_i, \kappa_i^*, t)$.

Based on these, the receding horizon control law for i-th vehicle at $t = \tau$ is defined as

$$\mathbf{u}_i(\tau) = \kappa_i^{RH}(\bar{\mathbf{x}}_i) = \kappa_i^*(0, \bar{\mathbf{x}}_i). \tag{28}$$

The running cost $\mathcal{L}_i(\cdot)$ has the form of

$$\mathcal{L}_i(\mathbf{x}_i, \mathbf{x}_{-i}^S, \mathbf{u}_i, \mathbf{y}_i^S) = \mathcal{L}_i^{gap}(\mathbf{x}_i, \mathbf{x}_{-i}^S) + \mathcal{L}_i^y(\mathbf{x}_i, \mathbf{y}_i^S) + \mathcal{L}_i^x(\mathbf{x}_i) + \mathcal{L}_i^u(\mathbf{u}_i), \tag{29}$$

where $-i$ represents index(es) of neighboring vehicle(s) following the notation of [7]. In case of the constant gap strategy,

$$\mathcal{L}_i^{gap}(\mathbf{x}_i, \mathbf{x}_{-i}^S) = ||\mathbf{e}_{i,i-1}^S||_{Q_{i,i-1}} + ||\mathbf{e}_{i,i+1}^S||_{Q_{i,i+1}}, \tag{30}$$

where $||x||_Q$ denotes a matrix weighted norm ($x^T Q x$). Similarly, using the varying gap strategy, the term can be represented as

$$\mathcal{L}_i^{gap}(\mathbf{x}_i, \mathbf{x}_{-i}^S) = ||\mathbf{e}_i^S||_{Q_i}. \tag{31}$$

In order to predict $\mathbf{x}_{-i}^S(\cdot)$ over the horizon, we use simple extrapolation strategy, which is described in Section 3.4.

$\mathcal{L}_i^y(\cdot)$ penalizes the tracking error, and can be defined as

$$\mathcal{L}_i^y(\mathbf{x}_i, \mathbf{y}_i^S(t)) = ||\mathbf{y}_i^S(t) - C_i^y(\mathbf{x}_i)||_{Q_y}, \tag{32}$$

where $C_i^y : \mathcal{X}_i \to \mathbb{R}^{n_i^y}$ maps the state vector into corresponding output signals. In order to track the reference velocity vector and heading in the spatial coordinate system, we use the following definition of $C_i^y(\mathbf{x}_i)$ in simulation:

$$C_i^y(\mathbf{x}_i) = \begin{bmatrix} R^{B\to S}\dot{\mathbf{x}}_i^B \\ \psi_i \end{bmatrix} \tag{33}$$

The term $\mathcal{L}_i^x(\cdot)$ is for remaining terms in the state vector that do not appear in the previous running costs, ϕ, θ, p, q and r, for example. It is noticeable that internal states, a, b, r_{fb}, c, and d, are not penalized, since they are not measurable, and related dynamics are well damped. Therefore, we can define $\mathcal{L}_i^x(\cdot)$ such that

$$\mathcal{L}_i^x(\mathbf{x}_i) = ||C_i^x \mathbf{x}_i||_{Q_x}, \tag{34}$$

where $C_i^x \in \mathbb{R}^{n_i^{x'} \times n_i^x}$, and $n_i^{x'}$ denotes the number of terms in the state penalized by $\mathcal{L}_i^x(\cdot)$. $\mathcal{L}_i^u(\cdot)$ penalizes input magnitudes,

$$\mathcal{L}_i^u(\mathbf{u}_i) = ||\mathbf{u}_i||_R, \tag{35}$$

with positive definite matrix $R \in \mathbb{R}^{n_i^u \times n_i^u}$.

Finally, the terminal penalty $V_i^f(\cdot)$ is defined by

$$V_i^f(\mathbf{x}_i, \mathbf{x}_{-i}^S, \mathbf{y}_i^S) = \begin{cases} \left\| \begin{matrix} \mathbf{e}_{i,i-1}^S \\ \mathbf{e}_{i,i+1}^S \\ \mathbf{y}_i^S - C_i^y(\mathbf{x}_i) \\ C_i^x \mathbf{x}_i \end{matrix} \right\|_{P_i} & \text{if constant gap strategy is used} \\ \left\| \begin{matrix} \mathbf{e}_i^S \\ \mathbf{y}_i^S - C_i^y(\mathbf{x}_i) \\ C_i^x \mathbf{x}_i \end{matrix} \right\|_{P_i} & \text{if varying gap strategy is used} \end{cases} \tag{36}$$

where $P_i \in \mathbb{R}^{(n^g + n_i^y + n_i^{x'}) \times (n^g + n_i^y + n_i^{x'})}$ is a positive definite matrix. Note that, for a follower that has two neighbors, the dimension of the gap error vector $n^g = 6$ when the constant gap strategy is used, and $n^g = 3$ when the varying

gap strategy is used. This terminal penalty plays several important roles in achieving guaranteed stability. In the next section, we discuss a design procedure of an MPC without time dependent terms $\mathcal{L}^{gap}(\cdot)$ and $\mathcal{L}^{y}(\cdot)$.

3.3 Design of Nonlinear MPC without Inter-agent Couplings

Due to pioneering research efforts in the 1990s, several design methodologies for stable MPC algorithms are now available [22]. In most of the stability proofs of MPC algorithms, the 'tail' of the value function of an FHOC problem plays a very important role, since it is known that, if one can approximate this term properly, the MPC based on FHOC problem realizes the virtues of infinite-horizon problems in stability and robustness. The work of Chen and Allgöwer [5] achieves this by taking advantage of the terminal inequality constraint and (virtual) terminal linear controller. Their technique for proving stability is one of the well-regarded among various finite-horizon based online optimization controllers, including a decentralized MPC algorithm [7]. However, it is reported that by introducing a terminal inequality constraint in the FHOC problem the numerical computation becomes slow [18], and sometimes the MPC structure nonrobust [12].

On the other hand, Jadbabaie *et. al.* [17] achieved a stable MPC algorithm by using the so-called control Lyapunov function (CLF) as a terminal cost without any terminal constraints. In this case, even though it is not easy to find a proper CLF for a given nonlinear system without conservatism, the scheme effectively minimizes the number of constraints subject to FHOC problem, which is quite important in practical implementation of an MPC algorithm.

In the remaining section, we describe the procedure to define a CLF for the nonlinear helicopter cruise model without inter-agent coupling and time dependent terms using semi-definite programming, which appears in [18].

First, we need to redefine helicopter cruise dynamics (Eq.(12)) without the spatial position vector part as follows

$$\frac{d}{dt}\begin{bmatrix}\mathbf{x}^D\\ \mathbf{x}^A\end{bmatrix} = \begin{bmatrix}A^D & A^A\\ A^{R_\omega^{B\to S}}(\mathbf{x}^A) & 0\end{bmatrix}\begin{bmatrix}\mathbf{x}^D\\ \mathbf{x}^A\end{bmatrix} + \begin{bmatrix}B\\ 0\end{bmatrix}\mathbf{u}, \tag{37}$$

where A in Eq.(12) is separated into A^A and A^D, and $A^{R_\omega^{B\to S}}(\mathbf{x}^A)$ is rearranged version of $R_\omega^{B\to S}$ (Eq.(6)) in corresponding order and dimension.

If we set upper and lower limits of ϕ and θ, then we can get bounds of those terms in $A_{R_\omega^{B\to S}}(\mathbf{x}^A)$. The operational limits of attitude variables are set to

$$-30^\circ \le \phi \le 30^\circ, \quad -40^\circ \le \theta \le 20^\circ. \tag{38}$$

The corresponding bounds of terms in $A^{R_\omega^{B\to S}}(\mathrm{x}^A)$ are

$$\begin{aligned}
-0.4195 \le \sin\phi\tan\theta &\triangleq p_1 \le 0.4195\\
-0.8391 \le \cos\phi\tan\theta &\triangleq p_2 \le 0.3640\\
0.8660 \le \cos\phi &\triangleq p_3 \le 1\\
-0.5 \le -\sin\phi &\triangleq p_4 \le 0.5\\
-0.6527 \le \sin\phi\sec\theta &\triangleq p_5 \le 0.6527\\
0.8660 \le \cos\phi\sec\theta &\triangleq p_6 \le 1.3054
\end{aligned}$$

Now we are ready to convert the system matrix in Eq.(37) into an affine parameter varying matrix [10] such that

$$A(p(t)) = A_0^p + \sum_{i=1}^{6} p_i(t) A_i^p, \tag{39}$$

where A_i^p is a constant matrix that has only one 1 on the corresponding entry, and zeros otherwise. A_0^p is the matrix that has constant terms in the system matrix of Eq.(37). Finally, the above parameter varying matrix can be represented by a polytopic model

$$A(t) \in \mathrm{Co}\{A_1^v, A_2^v, ..., A_{n_v}^v\}, \tag{40}$$

where the set $\mathrm{Co}\{\cdot\}$ denotes the set that includes all possible convex combination of its vertex elements. The conversion from Eq.(39) to Eq.(40) can be done by the function `aff2pol` in LMI Toolbox [10], and it results in a polytopic model with 64 vertices.

For a given weighting matrices (only for internal states, heading, and attitude variables extracted from Eq.(29)), the minimum upper bound of the value function is the optimal value of the following convex optimization problem [18]:

$$\min \mathrm{tr}(Z) \tag{41}$$

$$\begin{aligned}
&Y > 0\\
&\begin{bmatrix} YA_i^{vT} + A_i^v Y - BX - X^T B^T & YQ^{1/2} & X^T R^{1/2} \\ Q^{1/2}Y & -I & 0 \\ R^{1/2}X & 0 & -I \end{bmatrix} < 0\\
&\begin{bmatrix} Z & I \\ I & Y \end{bmatrix} > 0\\
&Y^T = Y,\ Z^T = Z,
\end{aligned} \tag{42}$$

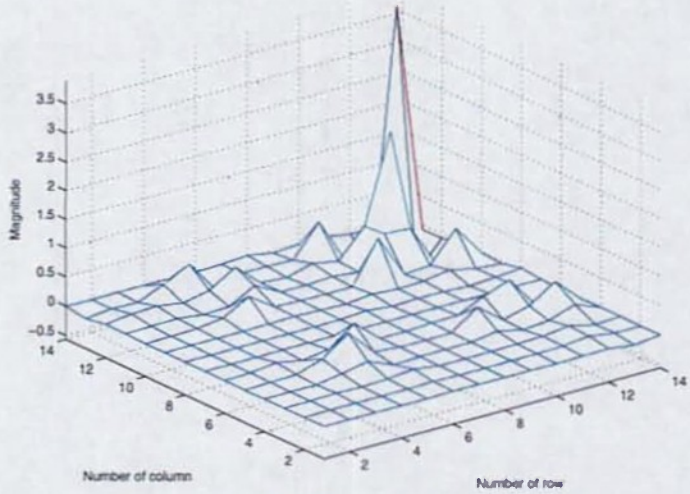


Figure 9: Magnitudes of elements in matrix P

where Z is a slack variable, $Y = P^{-1}$, and $X = KY$ are the change of variables. From this result, the terminal cost (for the system without external time-dependent signals) is defined by $V^f(\mathbf{x}) = \mathbf{x}^T P \mathbf{x}$.

Due to the size of the given cruise dynamics, the dimension of the above convex optimization problem is prohibitively huge. The problem has $2(n^x)^2 - (n^x - 1)(n^x - 2) + n^x n^i$ (292 in our case) variables and 64+1+1 LMI constraints whose dimensions are $(2n^x + n^u)^2$, $(2n^x)^2$, and $(n^x)^2$, respectively. Most of contemporary personal computers are based on a 32 bit structure, and their maximum allowable memory block size is limited to four giga bytes, which is marginal for our problem. Using LMITOOL [11] and SeDuMi [35], we tried to solve the full-scale problem, but a solution was not complete even after 96 hours. Instead, we sampled 32 vertices from 64 vertices in the definition of the polytopic system, and used them for CLF computation. In this case, the solver converges after 15 hours. The magnitudes of elements in the obtained matrix are shown in Figure 9.

For the given CLF $V^f(\mathbf{x})$, it is known that there exist $r \in \mathbb{R}_+$ such that

$$\min_{\mathbf{u}} (\dot{V}^f(\mathbf{x}) + \mathbf{x}^T Q \mathbf{x} + \mathbf{u}^T R \mathbf{u}) \leq 0 \quad \text{for } \mathbf{x} \in \Omega_r, \tag{43}$$

where $\Omega_r = \{\mathbf{x} \in \mathbb{R}^{n_D + n_A} | V^f(\mathbf{x}) \leq r\}$, and n_D and n_A are dimensions of $\mathbf{x}^D$ and $\mathbf{x}^A$, respectively.

Let $\mathbf{x}^*(t; \bar{\mathbf{x}})$ be the optimal trajectory at $t \in [\tau, \tau + L]$ starting from

	$\phi(0)$ (degree)	$\theta(0)$ (degree)
IC1	20	30
IC2	20	-30
IC3	-40	30
IC4	-40	-30

Table 2: Initial conditions used for single helicopter simulations with the designed MPC controller

$\bar{\mathbf{x}} = \mathbf{x}(\tau)$. If $\mathbf{x}^*(\tau + L; \bar{\mathbf{x}}) \in \Omega_r$, then then the trajectory starting from $\bar{\mathbf{x}}$ converges to the origin under the RHC scheme [17].

To complete the design procedure, we need to choose a proper horizon length L so that we can have a sufficiently large invariant set that includes Ω_r. However, in high-dimensional systems, it is very difficult to compute an invariant set corresponding to an L analytically or numerically [17]. In our research, we set $L = 0.5$ (sec) based on several simulation results with various values of L. The sampling frequency is set to 50Hz ($\delta = 0.02$ sec) so that it is identical with that of our existing UAV control system.

In order to show the validity of the CLF computed from the sub-sampled polytopic set, computer simulations are performed with several initial conditions. As shown in Figure 10 ~ 12, the designed MPC scheme successfully stabilizes all the initial conditions in Table 2.

It is noticeable that all the states converge to the origin in spite of control input saturations in the beginning of simulations (Figure 13). Even though the design procedures we use here are originally for unconstrained MPC [17], the controller works well with input constraints in our case.

3.4 Interagent Information Structure and Communication

Provided that the inter-agent communication happens only one time per sampling instance,[5] a decentralized algorithm can be implemented with lower bandwidth communication channels, while a centralized setup requires high bandwidth communication channel on the central agent which solves the optimal control problem for every agent. However, in a decentralized setup, if inter-agent communications are required during a numerical iteration, the total amount of information transferred between agents can be more than that of a centralized case [27]. This scheme falls into a category

[5]This means that there is no communication while solving a local optimization problem.

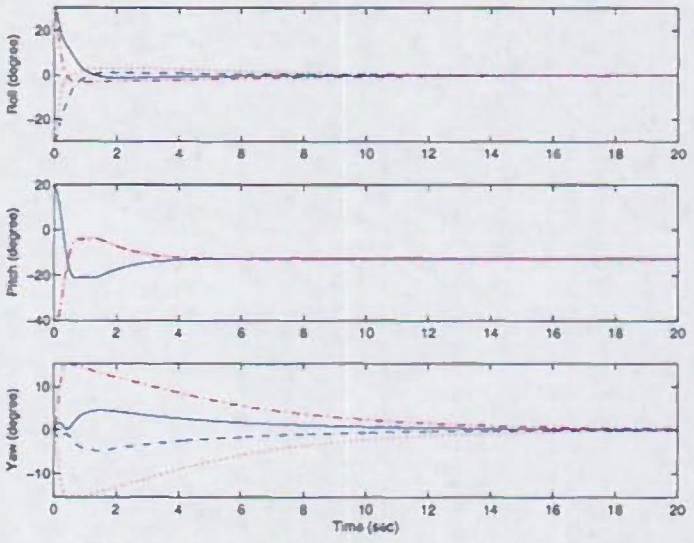


Figure 10: Attitude and heading responses. Note that the pitch angle should converge to the trim condition (-12.6 degree), not to zero. IC1: blue solid, IC2: blue dashed, IC3: red dash-dotted, IC4: red dotted

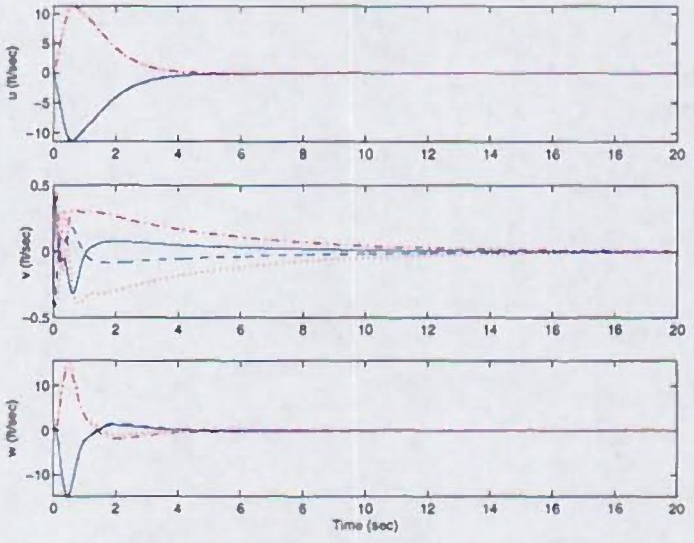


Figure 11: Body velocity responses, IC1: blue solid, IC2: blue dashed, IC3: red dash-dotted, IC4: red dotted

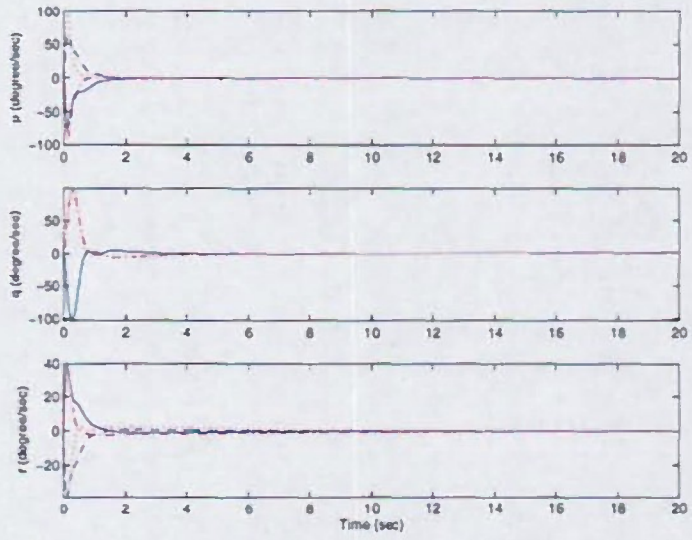


Figure 12: Body angular velocity responses, IC1: blue solid, IC2: blue dashed, IC3: red dash-dotted, IC4: red dotted

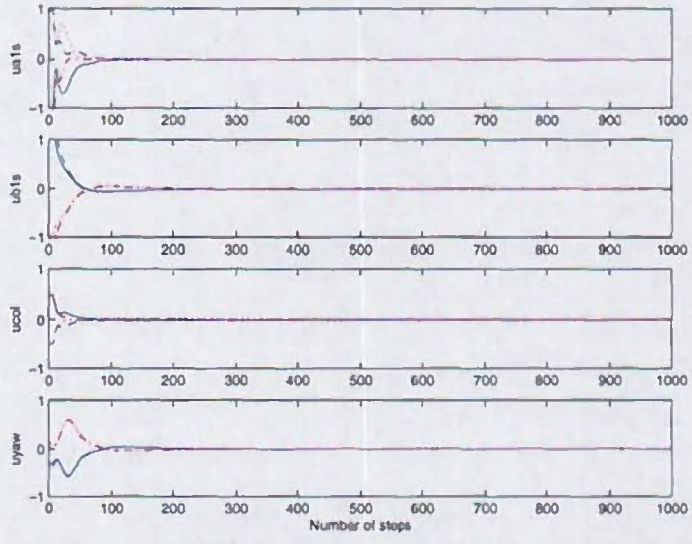


Figure 13: Control inputs, IC1: blue solid, IC2: blue dashed, IC3: red dash-dotted, IC4: red dotted

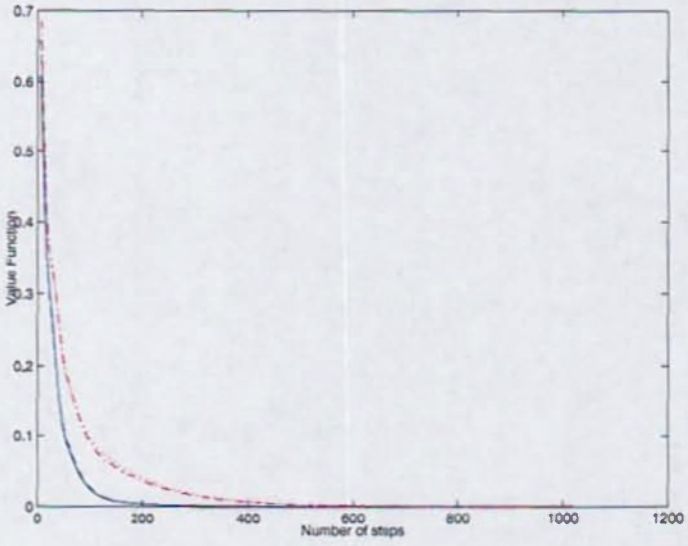


Figure 14: Values of , IC1: blue solid, IC2: blue dashed, IC3: red dash-dotted, IC4: red dotted

of algorithms using *cooperative iteration* [4]. Since this cooperative iteration needs stable and high bandwidth communication channels and strict synchronization between agents, it is more challenging to implement one on a hard real-time system. In the proposed setup, we use one inter-agent communication per every sampling instance, and there is no communication while solving an FHOC problem.

As shown in (29), the proposed MPC scheme requires the predicted trajectory of neighbors for $t \in [\tau, \tau + L]$ at every sampling time. In [7], neighboring agents interchange their predicted trajectories and use them as estimations of neighbors' trajectories. This appears to be a reasonable choice, but it requires higher communication bandwidth than the scheme that uses only current neighbors' states and extrapolates them for prediction. Moreover, few research is done about the cases that these predictions are not accurate due to external disturbances. The interagent information structure is one of the most controversial subjects in distributed model predictive research, e.g. [7, 19].

In our research, the transferred spatial positions of neighboring agents are extrapolated over the finite horizon. The predicted positions of neigh-

boring vehicles are represented by

$$\mathbf{x}^S_{-i}(\tau + t) = \mathbf{x}^S_{-i}(\tau) + \dot{\mathbf{x}}^S_{-i}(\tau)t \quad \text{for} \quad 0 \leq t \leq L. \tag{44}$$

As shown in the following section, acceptable performance can be achieved by this scheme. However, since the prediction error increases as the prediction horizon extends, the extension of the length of prediction horizon does not mean the enlargement of domain of attraction. In addition, the velocity information transferred to neighbors should be properly filtered so that the effects of noisy measurements can be minimized.

3.5 Simulations

The MPC algorithm for autonomous helicopter formations as formulated and described above was implemented in Matlab/Simulink environment. The core of the implementation involves a solution to the FHOC problem, and the following section is devoted to a discussion of numerical algorithms for FHOC problems.

3.5.1 Numerical Solver for Finite-Horizon Optimal Control Problem

In general, the nonlinear FHOC problem, a single player differential game from the viewpoint of the game theory, can be numerically solved in two ways: indirect and direct approaches. Indirect approaches utilize the necessary conditions given by the Pontryagin Minimum Principle. Then, it can be viewed as a multi-point boundary value problem, and an optimal solution is obtained by boundary value problem solvers like shooting methods and finite-element methods. However, these indirect methods are known to be very sensitive to their initial conditions, and as a result, lack robustness. See [28] and references therein. Therefore, it is not practical to use indirect methods in solving online optimization problems.

Direct methods, on the other hand, discretize continuous dynamics and cost functions using a high-order Runge-Kutta method [9, 28] or direct collocation [37], convert it into a finite-dimensional nonlinear optimization problem, and obtain an optimal solution through nonlinear programming techniques. These provide approximate solutions, but they are robust against arbitrary initial conditions, and optimal solutions with reasonable accuracy can be achieved using less intensive numerical procedures than indirect methods.

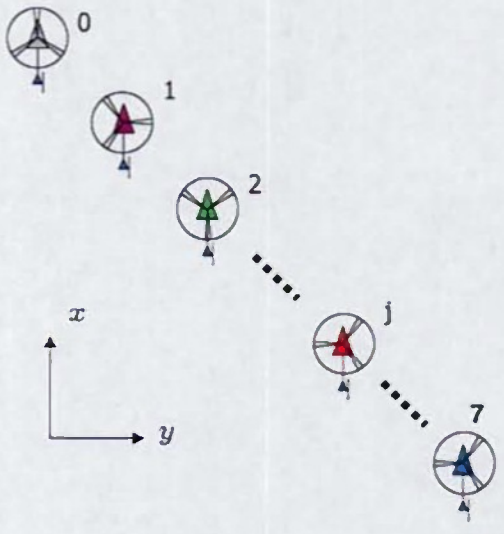


Figure 15: Right echelon formation with eight helicopters used in simulations

In the following simulations, we use the DynOpt package [9]. This package uses the 4th order Runge-Kutta method for discretization of the continuous-time dynamics and cost functions, and achieves an optimal solution using the sequential unconstrained minimization technique (SUMT). Since the package contains the SUMT algorithm and it is tightly integrated with discretization procedures, DynOpt allows for compact and versatile implementation. Although other solvers using direct method like NTG [24] and DIRCOL [37] require an external commercial nonlinear programming solver, it is worthwhile to use them in that they provide more user-friendly options and detailed error messages for debugging. In the case of NTG, it was reported that the package is used as an MPC engine in a hard real-time application [7].

3.5.2 Simulation Setup

Figure 15 shows the configuration of the right echelon formation [15] used in simulations. We chose this configuration because we want to investigate the propagation of disturbances through connected vehicles.

As shown in (29), the proposed MPC scheme requires the predicted trajectories of neighbors for $t \in [\tau, \tau + L]$ at every sampling time. In our

work, we adopt the scheme that interchanges the current position and velocity vectors of neighbors and extrapolates them over the prediction horizon. Although this information structure may prohibit us from using a long prediction horizon, since the prediction error may grow over prediction horizon, it allows us to minimize the interagent communication burden.

In the case of the constant gap strategy, gap vectors are defined as

$$l^r_{i,i-1} = \begin{bmatrix} 30 \\ -30 \\ 30 \end{bmatrix} \qquad l^r_{i,i+1} = \begin{bmatrix} -30 \\ 30 \\ -30 \end{bmatrix}, \tag{45}$$

and all units are in ft. Note that, even in the case using the varying gap strategy, vehicles in far edges, vehicle 0 and vehicle 7 in our configuration (Figure 15), using the constant gap strategy using the above constant relative gap vectors.

When we defined the FHOC problem (Eq.(25)), we assumed that there are constraints on states such that $\mathbf{x}_i \in \mathcal{X}_i$. In recent publications [12, 13], it is reported that the state-constrained MPC scheme is possibly not robust due to the discontinuity in the value function induced by state constraints. In accordance with this observation, we use only input constraints in our application, even though our FHOC problem solver allows state constraints in the formulation. We use the following admissible input set for all vehicles in our simulations

$$\mathcal{U}_i = \{\mathbf{u} \in \mathbb{R}^{n^u_i} | -1 \leq u^j \leq 1,\ 1 \leq j \leq n^u_i\}, \tag{46}$$

where u^j denotes the j-th element of the vector $\mathbf{u}$.

A final consideration here is the weighting on the attitude states, ϕ and θ (roll and pitch angles). It is well known in the field of aircraft control that the stabilization of attitude dynamics is a key to good controller design. This is due to the coupling between the translational dynamics and the attitude dynamics. In order to keep the attitude variation at a minimum, the terms related with ϕ and θ in (34) should be more heavily penalized than other terms.

3.5.3 Comparison Between Constant and Varying Gap Strategies

In order to compare disturbance attenuation performances of two types of gap error strategies, we exert negative longitudinal wind gust on the leading vehicle 0. The longitudinal acceleration induced by the wind gust is shown in Figure 16.

Figure 17 shows comparisons of the relative gap responses between the constant and the varying gap strategies. As shown here, the MPC algorithm successfully damps out the relative gap errors caused by the disturbance as they propagate into the formation irrespective of the gap error strategy type. This set of figures also illustrates an interesting feature of the varying gap strategy. When the varying gap strategy is used, the relative gap error between the Vehicle 0 and Vehicle 1 is lager than that of the constant gap strategy as shown in Figure 17(a). This is due to the nature of the varying gap strategy. When the Vehicle 0 approaches Vehicle 1, then Vehicle 1 can reduce the size of the gap error penalty very rapidly by moving toward Vehicle 2. In addition, the effects of reducing gap error penalty by the evasive motion of the Vehicle 2 is amplified through the structure of the varying gap strategy. Therefore, the relative gap between Vehicle 0 and Vehicle 1 can remain larger than in the constant gap strategy case. In other words, the varying gap strategy achieves more 'cooperation' from the neighboring vehicle(s) in terms of reducing the gap error penalty. Although the varying gap strategy shows poor performance in terms of the gap error between Vehicle 0 and Vehicle 1, Figure 18 shows that, in comparison with the constant gap strategy, the varying gap strategy can reduce the maximum velocity fluctuation of Vehicle 1, which is directly related to the passenger comfort. Also, the varying gap strategy shows superior performance from the point of view of the disturbance attenuation as the wind gust induced disturbance propagates through the formation (Figure 17(b) - Figure 17(f)).

The large gap error of the leader vehicle can be remedied by increasing the size of penalties on tracking errors. This is natural, since it is very important for a leader vehicle to maintain its velocity and heading, and the performance of the entire formation can depend on the tracking quality of the leader vehicle. For this reason, an active disturbance rejection scheme should be considered for vehicles at formation edges.

3.5.4 Performance with Heterogeneous Formations

In order to test our algorithm in a heterogeneous setup, we put virtual models from Section 2 in locations of 1, 2, 4, 5, and 6 in the right echelon formation (Figure 15). For Vehicles 0, 3, and 7, the original R-50 cruise model is used. The varying gap strategy is utilized in the following simulation.

As shown in Figure 19, the proposed algorithm successfully damps out external disturbance as it propagates through the formation.

In the case of mesh stability algorithm [30], the gap error induced by the leader motion is amplified between a normal vehicle and a more agile

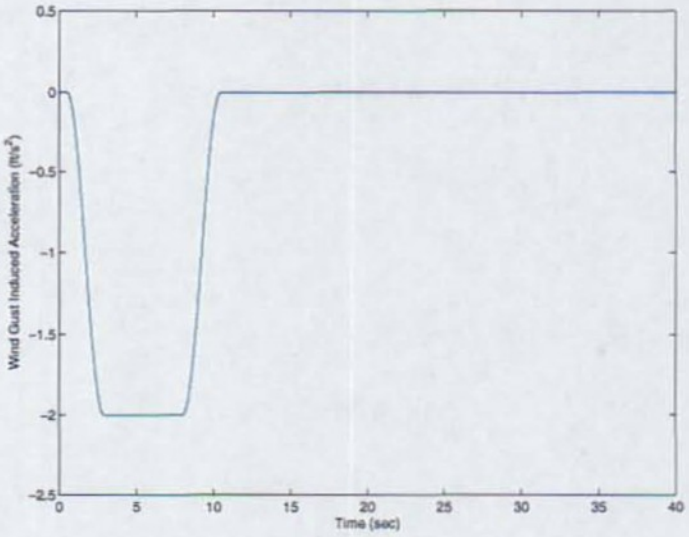


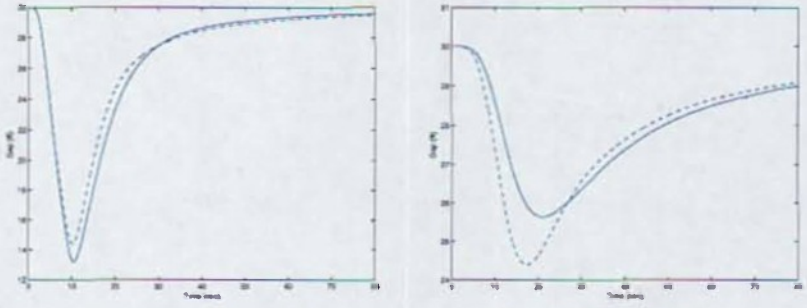
Figure 16: Disturbance induced by wind gust, negative x direction

vehicle. Since the algorithm uses position information of the leader as well as neighboring vehicles, and an agile vehicle tends to maintain the relative position from the leader even when the relative gap errors between neighbors become large, there exists a 'jump' in gap error propagation. However, in our MPC-based formulation, since vehicles share only reference velocities and heading, the dilemma of the global connection to the leader does not appear, and the space between vehicles can be safely maintained.

Figure 20 shows comparisons of gaps between homogeneous and heterogeneous formations. It is noticeable that the proposed algorithm shows comparable disturbance attenuation capability in heterogeneous formation.

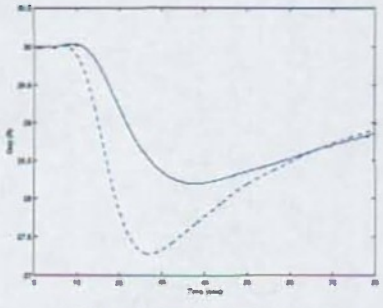
3.6 Discussion

The current problem of the proposed scheme is that the FHOC problem solver, DynOpt, is too slow for real-time applications. It takes about an hour to perform a 8-vehicle formation simulation for 80 seconds. Our algorithm will be tested with different solvers such as the gradient descent method [36]. We are optimistic because we already have performed successful MPC experiments using the gradient decent method [31]. The enhancement of the performance of our existing solver (using gradient decent method) is also now being pursued.

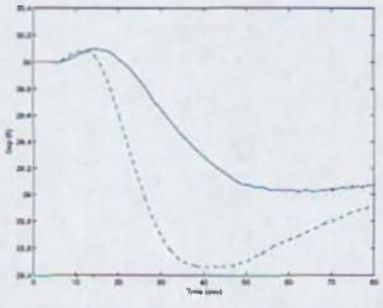


(a) Gap between Vehicle 0 and 1

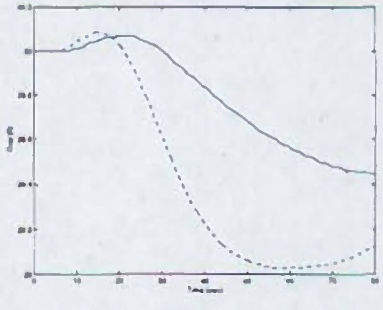
(b) Gap between Vehicle 1 and 2



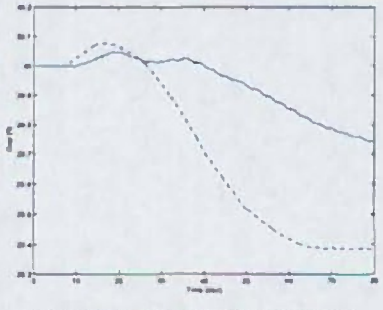
(c) Gap between Vehicle 2 and 3



(d) Gap between Vehicle 3 and 4



(e) Gap between Vehicle 4 and 5



(f) Gap between Vehicle 5 and 6

Figure 17: Comparison of gaps in x-direction, constant gap strategy: dashed lines, varying gap strategy: solid lines

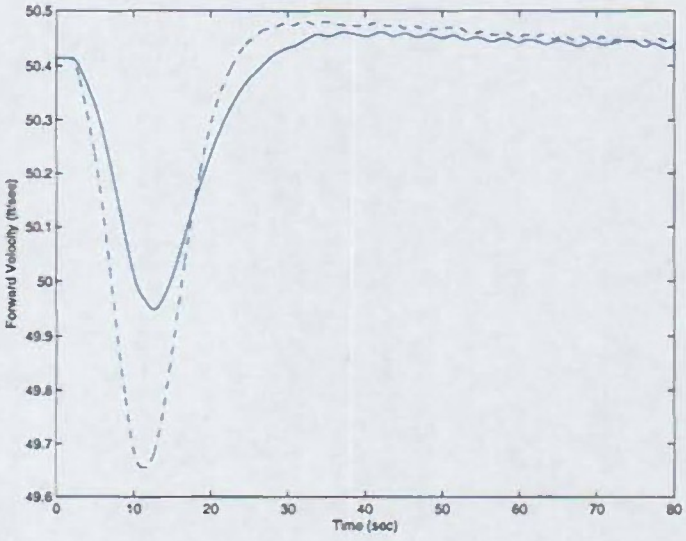


Figure 18: Forward velocity of the Vehicle 1 under the constant gap strategy (dotted), and the varying gap strategy (solid)

4 Hardware-In-the-Loop Simulation (HILS) System for BEAR Fleet

Hardware-in-the-Loop Simulator (HILS) allows embedded real-time modules to be tested in a simulated environment in closed-loop. The necessity of HILS system is emphasized especially in developing a complex system that requires high cost physical experiments. For example, the development of automotive control systems is also a well-known application area of HILS due to its difficulties on real car experiments [14].

Since the UAV system is extremely complicated, expensive, and safety-critical, it is natural to introduce a HILS system in its development cycle. In general, UAV experiments involve high risks, and a single typo in a huge code set can cause complete destruction of the system and/or injuries of ground staffs. Considering the risks involved in multi-UAV/UGV experiments like a formation flight, the entire development cycle may be severely delayed without sufficient testing on a HILS system.

The first version of the HILS system for BEAR fleet was implemented on real-time operating system (RTOS) VxWorks. In this version, a flight

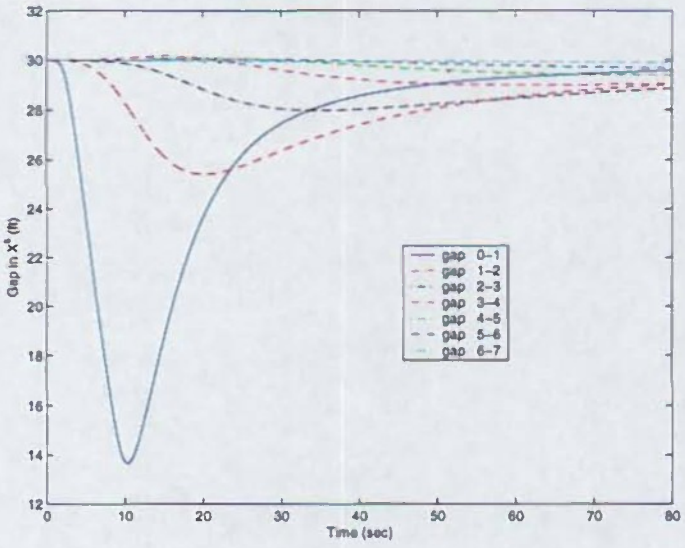
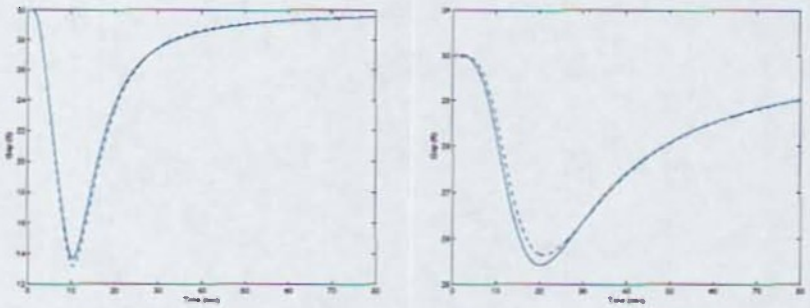
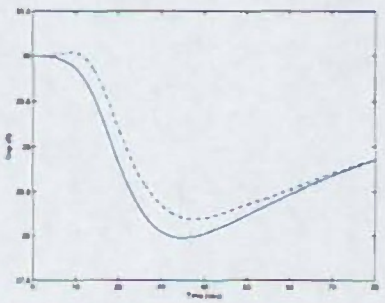


Figure 19: Gaps in x-direction, varying gap strategy under negative gust in longitudinal direction

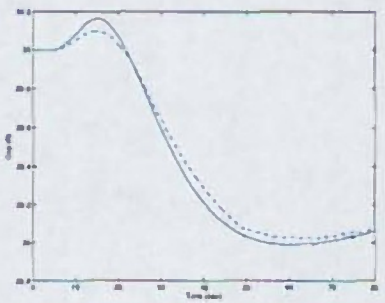


(a) Gap between vehicle 0 and 1

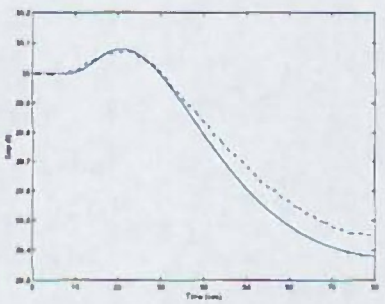
(b) Gap between vehicle 1 and 2



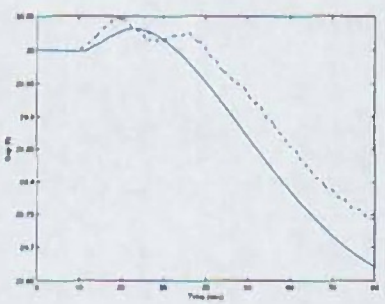
(c) Gap between vehicle 2 and 3



(d) Gap between vehicle 3 and 4



(e) Gap between vehicle 4 and 5



(f) Gap between vehicle 5 and 6

Figure 20: Comparison of gaps in x-direction, homogeneous formation: dashed lines, heterogeneous formation: solid lines

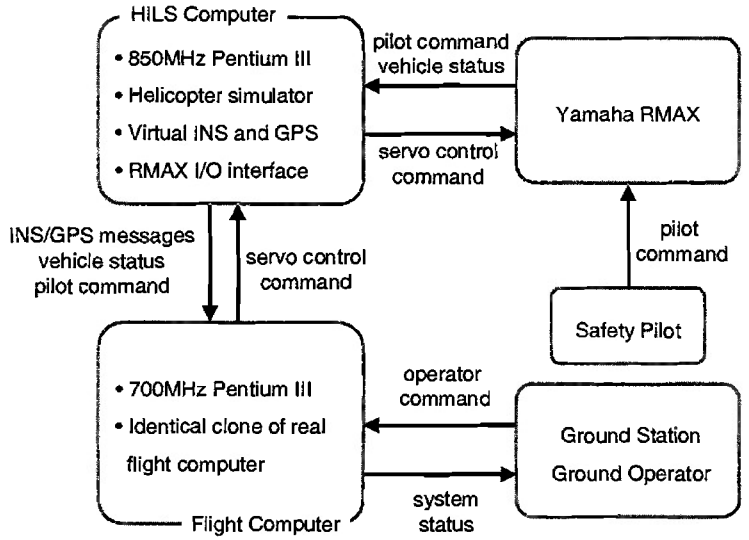


Figure 21: HILS system structure

controller for Yamaha RMAX was implemented using the time-based programming language, Giotto [16], and the effectiveness of Giotto was tested with the HILS system. However, since the actual control software for the RMAX was not available until 2004, only a simplified version of a controller was implemented. As a result, the VxWorks-based HILS system is not directly applicable to the control software running on our flight platforms. Moreover, it does not include a safety pilot and a ground station/operator in the loop.

Recently, in parallel with the development of the Ursa Maxima series based on Yamaha RMAX industrial helicopters, we developed a HILS system compatible with the existing navigation software in the BErkeley AeRobot (BEAR) program. In the following sections, we review the implementation of new version of our HILS system.

4.1 Hardware Structure

The hardware structure of HILS may vary depending on each systems's characteristics. In the case of development of a missile seeker system, the sensor systems are designed for a specific type of missile and the hardware structure is also a part of development. Hence the whole integrated flight system

System	Message	Description	Frequency (Hz)
GPS	PRTK	computed position, RTK	5
	PXY	Cartesian coordinate position	1
INS	M3	sensor status and navigation data	1
	M3512	delta velocity and delta attitude data	100
RMAX	RX	RMAX receiver data	40
	YACS	RMAX system status data	10

Table 3: INS, GPS, and RMAX messages used in BEAR avionics

including navigation sensors and processors should be tested in a simulated harsh environment. However, in the case of our UAV system, since it is made up of custom sensors and parts, and the hardware structure is not changed frequently, the test of entire avionics is not required. Rather, as a testbed of various complex scenarios, BEAR UAV systems may undergo a series of modifications in its software structure. Thus the most important feature to be included in the HILS system for the BEAR fleet is the verification ability of real-time software installed on the vehicles. Also, it is desirable to create a simulated situation as close to the real experiment as possible. In these contexts, we include a safety pilot in the loop as well as major navigation sensors like GPS and INS (Figure 21).

Pilot commands and RMAX vehicle status are relayed by Yamaha Attitude Control System (YACS) via serial ports. The HILS computer running on QNX4 generates various sensor outputs based on simulated helicopter dynamics and kinematics. The hardware components simulated in the HILS system are CMIGITS-II INS by Systron Donner and OEM3 GPS card by Novatel. Note that, as shown in Figure 21, our HILS system includes the vehicle, the safety pilot and ground station/operator so that it can create a situation that may happen in a real experiment.

4.2 Software Structure

The HILS system must provide all the INS/GPS and YACS messages used by the flight control system (FCS). Table 3 shows core messages required by FCS. Although the FCS software uses time-based and event-driven scheduling, these messages are purely time-based. Especially, the timing of the M3512 message should be kept precisely, since the basic scheduling of BEAR FCS software is depending on the arrival timing of this message. The software timers provided by RTOS QNX4 kernel are used for these timings, and

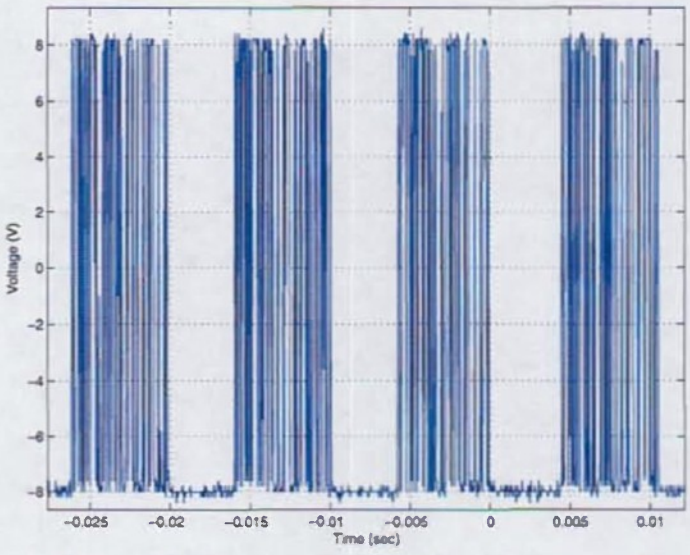


Figure 22: RS-232 pulse trains on INS M3512 channel

the timing performance of M3512 is measured by an oscilloscope. Figure 22 shows the measured RS-232 pulse trains of M3512. The average period between these pulse trains is approximately 10.24 ms (97.7 Hz) and timing jitter is less than 0.05 ms, which are small enough for our application. The entire software structure of the HILS system is shown in Figure 23. The HILS system consists of five processes: `server`, `ins`,`gps`, `rmax_io`, and `simulator`. `server`, which is not shown in the figure for simplicity, is the parent process of all other processes, and creates shared memories. This will be used as a gateway between the HILS control station and the HILS system in future development. The core computation parts of the above processes are ported from VxWorks version [16] to QNX4. Timing and shared memory structures are newly designed.

Process names are self-explanatory. `simulator` integrates helicopter dynamics and kinematics using Runge-Kutta 4-th order formula, and updates the shared memory `shm_sim_data` at 100 Hz. The process `gps` creates `pxy` and `prtk` messages based on simulation results and several coordinate conversions.[6] Likewise, `ins` creates M3512 and M3 messages using simulation

[6]For `prtk` message, we need a transformation from Cartesian coordinate to geodetic (Longitude-Latitude-Height, LLH) coordinate. Also, LLH coordinate to Earth-Centered-Earth-Fixed (ECEF) coordinate conversion is needed for calculation of local Cartesian

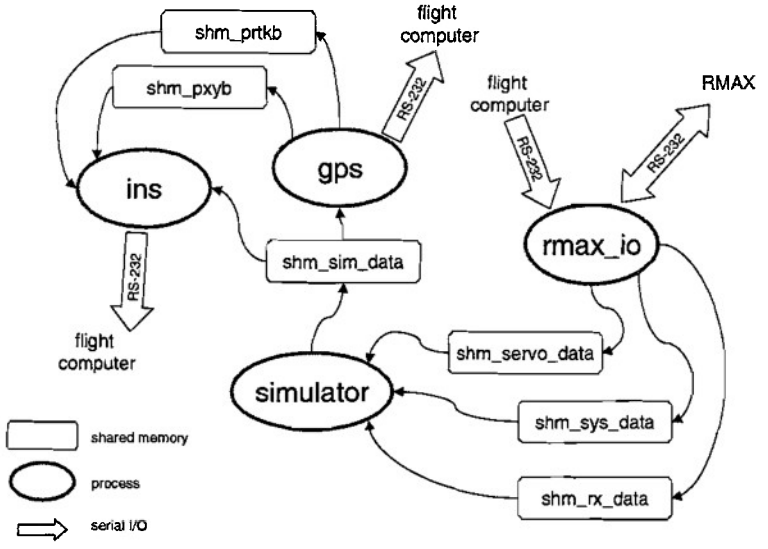


Figure 23: HILS system software structure

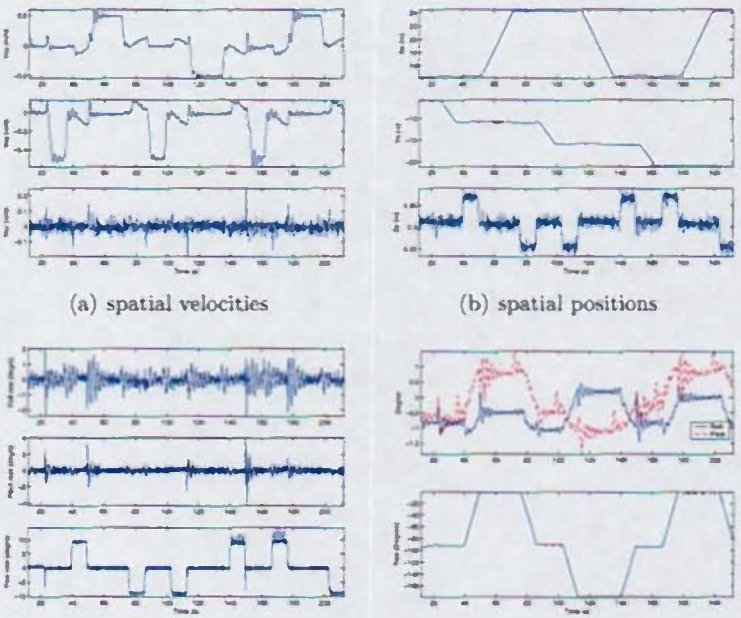
results and GPS conversion results. `rmax_io` deals with communications between the HILS system and YACS/FCS. In order to prevent simultaneous read/write access on a same shared memory, a mutually exclusive memory access structure is implemented using a memory access register in shared memory.

For more realistic sensor simulations, it is necessary to add proper measurement noises to the solutions from `simulator`. Based on [2] and data from experiments, properly scaled Gaussian noises are added to INS solutions. For GPS solutions, normally distributed uncorrelated random noises, whose standard deviation is equals to 0.02/3 (in meter scale) [25], are added to the solution.

4.3 Simulation Results

As an example of HILS system application, a waypoint navigation using Vehicle Control Language (VCL) [33] in batch mode is performed on the proposed HILS system. Figure 24 and Figure 25 show vehicle states during

origin.



(a) spatial velocities (b) spatial positions

(c) body rates (d) attitude and yaw angle

Figure 24: HILS simulation results during a waypoint navigation

the simulation and the vehicle trajectory in two-dimensional plane, respectively.

5 Conclusions and Future Work

In this ARO STIR W911NF-04-1-0448, the problem of autonomous helicopter formations is considered. A stable MPC-based controller for a single helicopter was implemented first, and then carefully designed inter-vehicle coupling terms were added in order to maintain safe space between helicopters. In Chapter 3, we showed the proposed algorithm successfully damps out exogenous disturbances via a series of simulations. The algorithm was also applied to a heterogeneous formation, and it showed a good attenuation property.

The simulations of the MPC-based formation flight scheme proposed throughout this report show that the algorithm involves more active and dynamic scenarios than the case of mesh stability [30]. It means that the experimental verification of the algorithm will be more challenging, and more

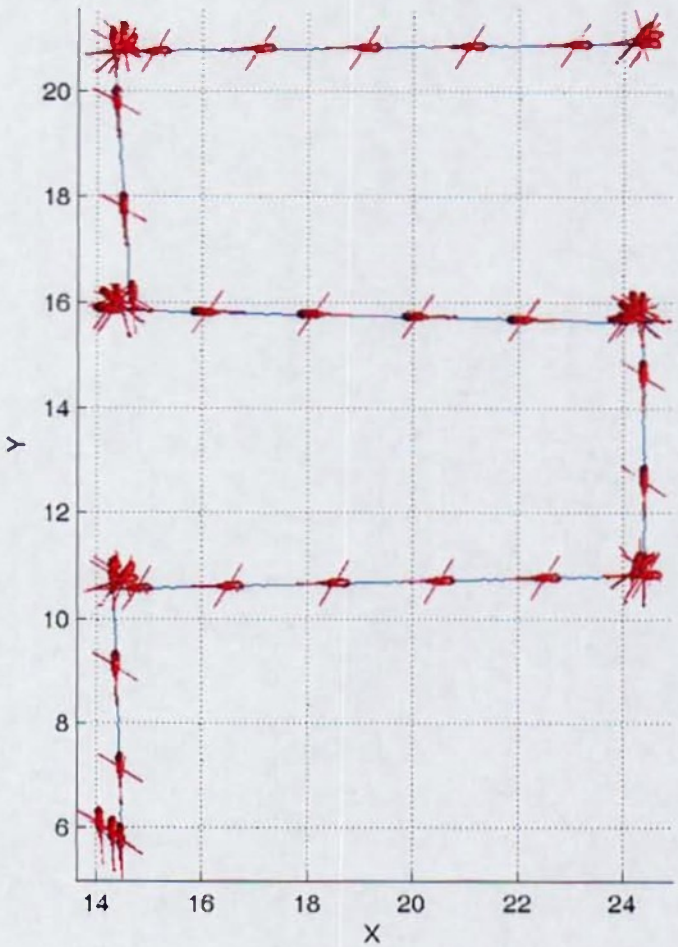


Figure 25: Waypoint navigation in the HILS environment

dangerous. In order to minimize the possibility of an unfortunate accident, newly implemented softwares should be strictly verified before the actual experiment, and all the experiments should be carefully designed considering the safety of ground crew. Our HILS system will play very important role in future development and experiments.

As the continuing effort of this project, we are now developing the concept of "formation manager" which is the high-level agent of the MPC controller enabling more dynamic and flexible autonomous formation flights. Also, the developed algorithms are now being implemented on real-time hardware and will be tested under HILS environment before the real-world experiments. All this work is currently supported by Phase I of the ARO STTR A05-T011 which ends in February 2006.

References

[1] `http://www.swarms.org/`.

[2] BEI Systron Donner Inertial Division. *C-MIGITS II Integrated INS/GPS User's Guide*, Feburary 2002.

[3] A. Bemporad and M. Morari. *Robust model predictive control: A survey*, pages 31–45. Lecture Notes in Computer Science. Springer-Verlag, 1999.

[4] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar. Distrubuted predictive control. *IEEE Control Systems Magazine*, pages 44–52, February 2002.

[5] H. Chen and F. Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1217, 1998.

[6] J. J. Craig. *Introduction to Robotics*. Addison-Wesley, 1989.

[7] W. B. Dunbar. *Distributed receding horizon control of multiagent systems*. PhD thesis, California Institute of Technology, 2004.

[8] W. D. Dunbar and W. Murray. Disributed receding horizon control with application to multi-vehicle formation stabilization. *Automatica*, (to appear).

[9] B. C. Fabien. Some tools for the direct solution of optimal control problems. *Advances in Engineering Software*, 29:45–61, 1998.

[10] P.l Gahinet, A. Nemirovski, A. J. Laub, and M. Chilali. *LMI Control Toolbox.* Mathworks, Inc., 1995. Now merged to Robust Control Toolbox.

[11] L. El Ghaoui, R. Nikoukhah, and F. Delebecque. LMITOOL: a package for LMI optimization. In *IEEE Conference on Decision and Control*, 1995.

[12] G. Grimm, M. J. Messina, S. E. Tuna, and A. Teel. Examples when nonlinear model predictive control is nonrobust. *Automatica*, 40:1729–1738, 2004.

[13] G. Grimm, M. J. Messina, S. E. Tuna, and A. Teel. Model predictive control: for want of a local control lyapunov function, all is not lost. *IEEE Transactions on Automatic Control*, 50:546–558, 2005.

[14] Herbert Hanselmann. Hardware-in-the-loop simulation testing and its integaration into a CACSD toolset. In *IEEE International Symposium on Computer-Aided Control System Design*, pages 152–156, September 1996.

[15] Headquarters, Department of the army. *Attack helicopter operations.* FM 1-112.

[16] B. Horowitz. *Giotto: A Time-Triggered Language for Embedded Programming.* PhD thesis, University of California, Berkeley, 2003.

[17] A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 37:1971–1978, 2001.

[18] Ali Jadbabaie. *Receding Horizon Control of Nonlinear Systems: A Control Lyapunov Function Approach.* PhD thesis, California Institute of Technology, 2000.

[19] T. Keviczky, F. Borrelli, and G. J. Balas. Hierarchical design of decentralized receding horizon controllers for decoupled systems. In *IEEE Conference on Decision and Control*, volume 2, pages 1592–1597, 2004.

[20] H. Kim. *Multiagent pursuit-evasion games: algorithms and experiments.* PhD thesis, University of California, Berkeley, 2001.

[21] M.V. Kothare, V. Banlakrishnan, and M. Morari. Robust constrained model predictive control using linear matrix inequalities. *Automatica*, 32:1361–1379, 1996.

[22] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Survey paper: Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[23] B. Mettler. *Identification modeling and characteristics of miniature rotorcraft.* Kluwer Academic Publishers, 2003.

[24] Mark B. Milam, Kudah Mushambi, and Richard M. Murray. A computational approach to real-time trajectory generation for constrained mechanical systems. In *IEEE Conference on Decision and Control*, volume 1, pages 845 – 851, 2000.

[25] NovAtel Inc. *MiLLennium GPSCard and Enclosures: Guide to Installation and Operation.* OM-20000016 Rev 3.

[26] A. G. Pant. *Mesh Stability of Formations of Unmanned Aerial Vehicles.* PhD thesis, University of California, Berkeley, 2002.

[27] R. L. Raffard, C. J. Tomlin, and S. P. Boyd. Distributed optimization for cooperative agents: Application to formation flight. In *IEEE Conference on Decision and Control*, pages 2453–2459, December 2004.

[28] Adam L. Schwartz. *Theory and Implementation of Numerical Methods Based on Runge-Kutta Integration for Solving Optimal Control Problems.* PhD thesis, University of California, Berkeley, 1996.

[29] P. J. Seiler. *Coordinated Control of Unmanned Aerial Vehicles.* PhD thesis, University of California, Berkeley, 2001.

[30] E. Shaw, H. Chung, J.K. Hedrick, and S. Sastry. Unmanned helicopter formation flight experiment for the study of mesh stability. In D. Grundel, R. Murphey, P. Pardalos, and O. Prokopyev, editors, *Advances in Cooperative Control and Optimization.* Kluwer Academic, 2005.

[31] D. Shim, H. Chung, H. J. Kim, and S. Sastry. Autonomous exploration in unknown urban environments for unmanned aerial vehicles. In *AIAA Conference on Guidance, Navigation and Control*, 2005.

[32] D. H. Shim. *Hierachical Flight Control System Synthesis for Rotorcraft-based Unmanned Aerial Vehicles.* PhD thesis, University of California, Berkeley, 2000.

[33] D. H. Shim, H. J. Kim, H. Chung, and S. Sastry. Multi-functional autopilot design and experiments for rotorcraft-based unmanned aerial

vehicles. In *20th Digital Avionics Systems Conference*, volume 1, pages 3C4/1–3C4/8, 2001.

[34] D. H. Shim, H. J. Kim, and S. Sastry. Decentralized nonlinear model predictive control of multiple flying robots in dynamic environments. In *IEEE Conference on Decision and Control*, volume 4, pages 3621–3626, 2003.

[35] Jos F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11-12:625–653, 1998.

[36] G. J. Sutton and R. R. Bitmead. Performance and computational implementation of nonlinear model predictive control on a submarine. In F. Allgoöwer and A. Zheng, editors, *Nonlinear Model Predictive Control*, volume 26 of *Progress in Systems and Control Theory*, pages 461–471. Birkhäuser, 2000.

[37] O. von Stryk. Numerical solution of optimal control problems by direct collocation. In J. Stoer R. Bulirsch, A. Miele and K.-H. Well, editors, *Optimal Control - Calculus of Variations, Optimal Control Theory and Numerical Methods*, International Series of Numerical Mathematics 111, pages 129–143. Birkhuser, 1993.